Adaptive Compressive Tracking via Online Vector Boosting Feature Selection

Qingshan Liu, Jing Yang, Kaihua Zhang*, Yi Wu

Jiangsu Key Laboratory of Big Data Analysis Technology, Nanjing University of Information Science and Technology, Nanjing, China

Abstract

Recently, the compressive tracking (CT) method [1] has attracted much attention due to its high efficiency, but it cannot well deal with the large scale target appearance variations due to its data-independent random projection matrix that results in less discriminative features. To address this issue, in this paper we propose an adaptive CT approach, which selects the most discriminative features to design an effective appearance model. Our method significantly improves CT in three aspects: Firstly, the most discriminative features are selected via an online vector boosting method. Secondly, the object representation is updated in an effective online manner, which preserves the stable features while filtering out the noisy ones. Finally, a simple and effective trajectory rectification approach is adopted that can make the estimated location more accurate. Extensive experiments on the CVPR2013 tracking benchmark demonstrate the superior performance of our algorithm compared over state-of-the-art tracking algorithms.

Keywords:
tracking by detection, compressive tracking, feature template, model update

1. Introduction

Object tracking is a fundamental problem in computer vision with numerous applications such as motion analysis, surveillance, autonomous robots,etc, and much process has been witnessed in recent years [2]. However, it remains

*Corresponding author. Phone number: 086-13851581017
Email address: zhkhua@gmail.com (Kaihua Zhang)

a challenging task due to the factors like illumination changes, partial occlusion, deformation, as well as viewpoint variation, to name a few [3]. To well handle these factors, an effective appearance model is of great importance, in which numerous design schemes have been proposed [4, 5, 6, 7, 8, 9, 10, 11, 1], which can be categorized into either generative models or discriminative ones.

Generative models learn an appearance model with the object information, which is used to search for the object with the minimum reconstruction error within a certain region. Adam et al. [12] represent the target appearance with the intensity histograms of multiple fragments that can be efficiently computed by integral images. Ross et al. [13] present a tracking method that incrementally learns a low-dimensional subspace representation, which can effectively adapt to the target appearance changes. Mei and Ling [14] treat tracking as a sparse representation problem, in which the target location is determined by solving an ℓ_1 minimization problem. Bao et al. [15] utilize the accelerated proximal gradient approach to efficiently solve the ℓ_1 minimization problem for visual tracking. In [8], Kwon and Lee propose a visual tracking decomposition method that combines multiple observation and motion models for robust visual tracking, which has been further extended to search for appropriate trackers by the Markov Chain Monte Carlo sampling method [9]. Zhang et al. [7] formulate the tracking task as a multi-task sparse learning problem. In [6], Jia et al. formulate the object appearance as sparse codings of the local image patches with their spatial layout. In [7], Zhong et al. propose a collaborative tracking algorithm that combines a sparsity-based discriminative classifier and a sparsity-based generative model. Wang et al. [16] present a least soft-threshold squares algorithm that models the image noise with the Gaussian-Laplacian distribution.

Discriminative models learn a binary classifier to distinguish the target from its surrounding background. Avidan [17] first proposes to utilize a support vector machine classifier for visual tracking. In [18], an online discriminative feature selection technique is proposed to extract the most discriminative features to separate the target from the background. Grabner et al. [4] proposes an online boosting method to select features for visual tracking. Babenko et al. [10] formulate the tracking task as a multiple instance learning (MIL) problem, and propose an online MIL boosting method that selects features to design an appearance model. Zhang and Song [19] further extend the MIL tracking method by considering the sample importance. Kalal et al. [11] integrate a re-detection module into tracking that can restart tracking after the target reappears when it is completely occluded or missing from

the scene. Hare et al. [20] exploit the constraints of the predicted outputs with a kernelized structured SVM classifier, which achieves favorable results on the CVPR2013 tracking benchmark [21]. Henriques et al. [22] propose a fast tracking algorithm which explores the circulant structure of the kernel matrix in the SVM classifier that can be efficiently computed by the fast Fourier transform algorithm. Zhang et al. [1] propose a real-time compressive tracking (CT) algorithm that employs a very sparse random matrix to achieve a low-dimensional image representation. In [23] Zhang et al. further reduce the computational complexity of CT with a coarse-to-fine strategy. Song [24] improves the performance of CT by introducing informative feature selection strategy.

Recently, Wu et al. [21] release the CVPR2013 tracking benchmark, which contains 50 challenging sequences ($\sim$ 26000 frames), most of which suffer large scale target appearance variations. Results of 29 tracking algorithms are reported including most above mentioned tracking algorithms. Although CT is very efficient that runs over 60 frames per second (FPS), its success rate of one-pass evaluation (OPE) is only 30.6%. We claim that the unfavorable performance of CT is due to its data-independent random projection matrix that can only yield fixed feature templates, which cannot adapt the large scale target appearance variations well. In this paper, we propose an adaptive CT method which selects the most discriminative patches to construct the feature templates via a novel online vector boosting method. Furthermore, we adopt a new model update mechanism that can preserve the stable features while avoiding the noisy ones, thereby effectively alleviating the drift problem caused by online model update. Finally, we propose a very simple trajectory rectification method that makes the finally estimated location more accurate. Extensive evaluations on the CVPR2013 tracking benchmark [21] demonstrate the proposed algorithm performs favorably against state-of-art methods in terms of efficiency, accuracy and robustness, and especially the proposed algorithm outperforms CT by a large margin (the success rate of OPE of the proposed method is 50.4% vs. 30.6% for CT).

The key contributions of the proposed algorithm are summarized as follows:

- To the best of our knowledge, it is the first time to explore online vector boosting [25] feature selection method for visual tracking, in which the selected features can well adapt to the target appearance variations.
- A novel trajectory rectification strategy is proposed that can be readily

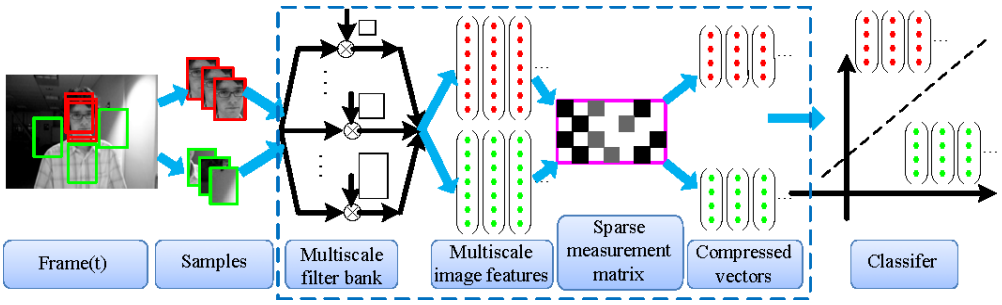

Figure 1: The main components of CT

extended to other tracking algorithms to ensure more accurate and stable tracking results.

- Our tracker achieves favorable results with 50.4% in success plots and 71.4% in precision plots, which ranks 1st in the CVPR2013 tracking benchmark [21], showing the power of the simple Haar-like features.

2. Compressive Tracking

The idea of CT [1] is motivated by the compressive sensing theory [26, 27] in which the random projections of a sufficiently high dimensional feature vector contain enough information to reconstruct the original high-dimensional one. The main components are illustrated by Figure 1. First, each sample is represented by a high-dimensional multiscale vector via convolving each patch with some rectangle filters. Then, the vector is projected onto a low-dimensional space with a very sparse random projection matrix that satisfies the restricted isometry property (RIP) of the compressive sensing theory. The original high-dimensional feature vectors can well discriminate the target from its local background while the high efficiency is achieved by the very sparse random matrix, and thereby CT performs well on some challenging sequences in terms of both efficiency and accuracy.

CT employs a very sparse random matrix $\mathbf{R} \in \mathbb{R}^{n\times(wh)^2}$ to project the high-dimensional vector $\mathbf{x}$ onto a low-dimensional vector $\mathbf{v} \in \mathbb{R}^n$

$$\mathbf{v} = \mathbf{R}\mathbf{x}, \tag{1}$$

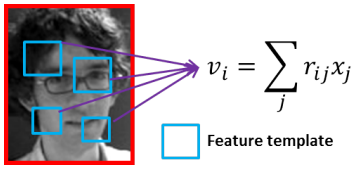

Figure 2: Each compressed feature is constructed by several feature templates

where the entry of $\mathbf{R}$ is represented by

$$r_{ij} = \sqrt{\rho} \times \begin{cases} 1 & \text{with probability } \frac{1}{2\rho} \\ 0 & \text{with probability } 1 - \frac{1}{\rho} \\ -1 & \text{with probability } \frac{1}{\rho}. \end{cases} \tag{2}$$

where $\rho = \frac{(w \times h)^2}{4}$ with w and h representing width and height of the object size, respectively.

In (1), the i-th feature v_i in the compressed vector $\mathbf{v}$ can be represented as

$$v_i = \sum_j r_{ij} x_j. \tag{3}$$

Figure 2 illustrates that v_i in (3) is constructed by several feature templates, whose sizes and locations are set randomly and fixed during tracking. Although this random selection strategy is simple and efficient, it has some limitations that makes CT perform unfavorably when the target appearance varies much: First, the feature templates may select noninformative features when they fall into the textureless regions. Second, the fixed templates cannot adapt to the target appearance variations well. In the following section, we will propose an adaptive CT that can deal with these issues well.

3. Adaptive Compressive Tracking

3.1. Algorithm overview

Figure 3 illustrates the basic flow of our tracking algorithm that is summarized in Algorithm 1, which mainly consists of three steps. First, we construct a set of positive and negative feature template bags $\{\mathcal{B}_i^+, \mathcal{B}_i^-\}_{i=1}^c$, in which each bag $\mathcal{B}_i = \{\mathbf{z}_{ij}\}_{j=1}^n$ contains n rectangle feature templates, of which each template $\mathbf{z}_{ij}$ represents a vectorized image patch inside the blue

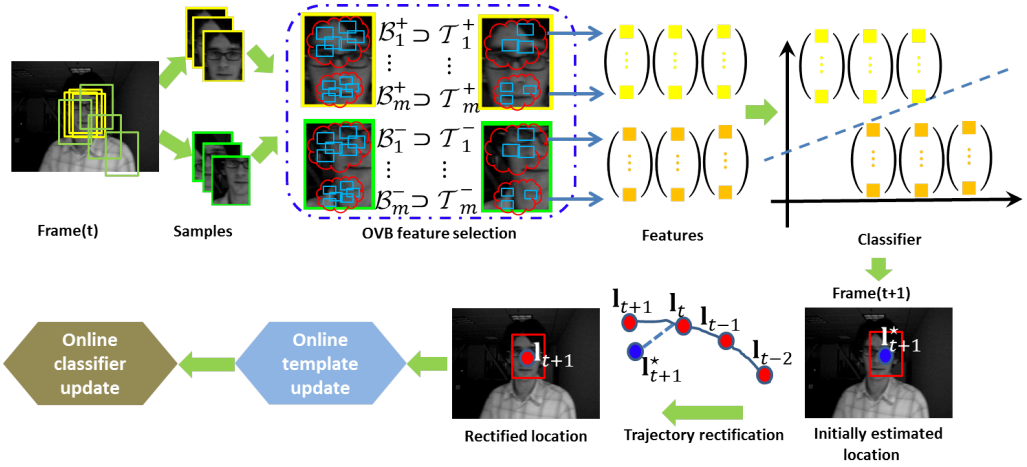

Figure 3: Flowchart of ACT. OVB is short for online vector boosting (refer to Algorithm 2).

rectangle, and then we select k templates via an online vector boosting feature selection strategy, which constructs the feature template bags $\mathcal{T}_i \subset \mathcal{B}_i$. Second, to take into account the target appearance variations over time, we exploit an online template update strategy that preserves the stable feature templates while replacing the ones with remarkable changes by a linear combination of former and current templates. Finally, when the score of the maximum classifier confidence for the estimated tracking location is lower than a threshold Θ, which indicates that the estimation is inaccurate, then we employ a trajectory rectification strategy that utilizes the former tracking motion information to predict the current tracking location.

3.2. Online vector boosting feature selection

As illustrated by Figure 2, the templates in CT [1] are constructed by several patches with random locations and sizes, of which the size of each patch ranges from 1×1 to $w \times h$ pixels, where w and h represent width and height of the object, respectively. However, a too small patch is vulnerable to the noisy small appearance variations while a too large one cannot distinguish the most likely target from its neighboring counterparts due to its large support. To handle this problem, we constrain the width and height of the feature template by $2 < t_{w_i} < round(w/2)$, $2 < t_{h_i} < round(h/2)$. Furthermore, to take into account multiscale appearance information, we set the patches in the same bag to the same size while the patches in different

Algorithm 1 Adaptive Compressive Tracking (ACT)

Input: The t-th image frame

1: Sample a set of image patches in $D^{\gamma} = \{\mathbf{p} | ||\mathbf{l}_t(\mathbf{p}) - \mathbf{l}_{t-1}|| < \gamma\}$ where $\mathbf{l}_{t-1}$ is the tracking location at the $(t-1)$-th frame, and extract features with the feature template $\mathcal{T}$

2: Apply the classifier $H(\cdot)$ in (15) to each feature vector, and get the maximum confidence score $conf$

3: **if** $conf < \Theta$ **then**

4: Rectify the tracking location $\mathbf{l}_t$ via (18)

5: **else**

6: Find the tracking location $\mathbf{l}_t$ via maximizing the classification score

7: **end if**

8: Sample two sets of image patches $D^{\alpha} = \{\mathbf{p} | ||\mathbf{l}_t(\mathbf{p}) - \mathbf{l}_t|| < \zeta\}$ and $D^{\alpha,\beta} = \{\mathbf{p} | \alpha < ||\mathbf{l}_t(\mathbf{p}) - \mathbf{l}_t|| < \beta\}$ with $\zeta < \alpha < \beta$

9: Update the feature template bags $\mathcal{B}$ and the classifier parameters

Output: Tracking location $\mathbf{l}_t$

bags own varying sizes. Next, we will introduce our OVB feature selection method that can select the most discriminative feature templates from each bag to design a strong classifier.

As illustrated by Figure 4, providing the positive and negative feature template bags $\mathcal{B}_i^+, \mathcal{B}_i^-, i = 1, \ldots, c$, we define a margin between them that is the sum of Euclidean distance between the average positive and negative feature vectors in each template

$$margin = \sum_{i=1}^{c} margin_i, \tag{4}$$

where $margin_i$ is defined as

$$\begin{aligned} margin_i &= |\overline{\mathbf{Z}_i^+} - \overline{\mathbf{Z}_i^-}|_2^1 \\ &= \sqrt{\overline{\mathbf{Z}_i^+}^{\top}\overline{\mathbf{Z}_i^+} - \overline{\mathbf{Z}_i^+}^{\top}\overline{\mathbf{Z}_i^-} - \overline{\mathbf{Z}_i^-}^{\top}\overline{\mathbf{Z}_i^+} + \overline{\mathbf{Z}_i^-}^{\top}\overline{\mathbf{Z}_i^-}} \\ &= \sqrt{2n - 2\sum_{j=1}^{n} \overline{\mathbf{z}_{ij}^+}^{\top}\overline{\mathbf{z}_{ij}^-}} \end{aligned} \tag{5}$$

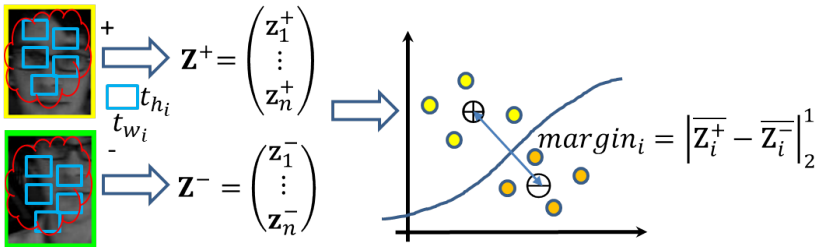

Figure 4: Illustration of the defined margin for the features in the i-th bag $\mathcal{B}_i$. t_{w_i}, t_{h_i} denote the width and height of the rectangle template in the i-th bag, respectively. $\mathbf{Z}_i^+, \mathbf{Z}_i^-$ denote the corresponding image representations of positive and negative samples, respectively, in which $\mathbf{z}_{i,1,\ldots,n}^+$ and $\mathbf{z}_{i,1,\ldots,n}^-$ denote the feature templates that are the normalized image patch vectors in the blue rectangles. $\overline{\mathbf{Z}_i^+}$ and $\overline{\mathbf{Z}_i^-}$ denote the average image representations of the positive and negative samples, respectively.

where $\overline{\mathbf{z}_{ij}^+}$ and $\overline{\mathbf{z}_{ij}^-}$ denote the j-th normalized feature templates in the i-th bag of the positive and negative samples, respectively.

It is easy to verify that $\overline{\mathbf{z}_{ij}^+}^\top \overline{\mathbf{z}_{ij}^-} \leq \overline{\mathbf{z}_{ij}^+}^\top \sum_{j=1}^n \overline{\mathbf{z}_{ij}^-}$, so we have

$$margin_i = \sqrt{2n - 2\sum_{j=1}^{n} \overline{\mathbf{z}_{ij}^+}^\top \overline{\mathbf{z}_{ij}^-}} \geq \sqrt{2n - 2\sum_{j=1}^{n} \overline{\mathbf{z}_{ij}^+}^\top \sum_{j=1}^{n} \overline{\mathbf{z}_{ij}^-}} = \mathcal{J}(\overline{\mathbf{z}_{i1}}+, \ldots, +\overline{\mathbf{z}_{in}}), \tag{6}$$

where $\mathcal{J}$ is the lower bound of the margin function $margin_i$. For each sample $\mathbf{p}$, its image representation in the i-th bag is $\mathcal{B}_i(\mathbf{p}) = \{\mathbf{z}_{ij}(\mathbf{p})\}_{j=1}^n$, and we utilize the template center bag to robustly represent each class as $\overline{\mathcal{B}_i} = \{\overline{\mathbf{z}_{ij}}\}_{j=1}^n$ (See Figure 4). Our objective is to select a subsect of feature templates $\{\overline{\mathbf{z}_{ij}}\}_{j=1}^k$ from bag $\overline{\mathcal{B}_i}$ that maximizes the margin function $margin_i$, which can be readily achieved by maximizing its lower bound $\mathcal{J}$

$$\{\overline{\mathbf{z}_{i1}}, \ldots, \overline{\mathbf{z}_{ik}}\} = \arg\max_{\{\overline{\mathbf{z}_{i1}}, \ldots, \overline{\mathbf{z}_{ik}}\} \subset \overline{\mathcal{B}_i}} \mathcal{J}(\overline{\mathbf{z}_{i1}}+, \ldots, +\overline{\mathbf{z}_{ik}}). \tag{7}$$

The vector boosting algorithm in [25] relies on the special case of the exponential loss function of AdaBoost, and thus cannot be readily adapted to solve the above problem. Now, we present the proposed novel online vector boosting algorithm that can readily address the above problem. Our method is motivated by the algorithm in [28] that takes the statistical view of boosting, which tries to optimize a specific objective function $\mathcal{L}$ by sequentially

optimizing the following criterion

$$(h_j, \alpha_j) = \arg\max_{h_j \in \mathcal{H}, \alpha} \mathcal{L}(H_{j-1} + \alpha h_j), \tag{8}$$

where $H_{j-1}(\mathbf{p}) : \Omega \rightarrow \mathbb{R}$ is a strong classifier that is the sum of the first $j-1$ weak classifiers $h_i, i = 1, \ldots, j-1$ and $\mathcal{H}$ is the set of all possible weak classifiers.

The proposed algorithm is an extension of the optimization problem in (8) in which both the outputs of its weak classifiers and final output are vectors rather than scalars. At all time we maintain n candidate weak classifiers in which the j-th weak classifier is defined as

$$\mathbf{h}_{ij}(\mathbf{p}) = \overline{\mathbf{z}_{ij}}(\mathbf{p}). \tag{9}$$

To update the classifier, we first update a subset of weak classifiers in parallel via an online feature template update strategy (refer to Section 3.3), and then we choose k weak classifiers $\mathbf{h}_{ij}$ from the candidate pool sequentially by maximizing the lower bound $\mathcal{J}$ in (7)

$$\mathbf{h}_{ij} = \arg\max_{\mathbf{h}_{ij} \in \{\overline{\mathbf{z}_{i1}}, \ldots, \overline{\mathbf{z}_{in}}\}} \mathcal{J}(\mathbf{H}_{i(j-1)} + \mathbf{h}_{ij}), \tag{10}$$

where $\mathbf{H}_{i(j-1)} = \sum_{l=1}^{j-1} \mathbf{h}_{il}$. Algorithm 2 summarizes the main steps of the proposed online vector boosting method.

At last, we concatenate all the selected feature templates in all bags to yield a high-dimensional multiscale image representation $\mathbf{x} = (\mathbf{h}_{11}^\top, \ldots, \mathbf{h}_{1k}^\top, \ldots, \mathbf{h}_{c1}^\top, \ldots, \mathbf{h}_{ck}^\top)^\top \in \mathbb{R}^{k \sum_{i=1}^{c} t_{w_i} t_{h_i} \times 1}$. We then utilize an orthogonal matrix $\mathbf{S} \in \mathbb{R}^{c \times k \sum_{i=1}^{c} t_{w_i} t_{h_i}}$ to project $\mathbf{x}$ onto a c-dimensional feature space

$$\mathbf{v} = \mathbf{S}\mathbf{x}, \tag{11}$$

where the entry of $\mathbf{S}$ is denoted as

$$s_{ij} = \frac{1}{\sqrt{k t_{w_i} t_{h_i}}} \times \begin{cases} 0 & j < (i-1) k t_{w_i} t_{h_i}, j > i k t_{w_i} t_{h_i}, \\ \pm 1 & \text{with equal probability}, \end{cases} \tag{12}$$

and the i-th entry of $\mathbf{v}$ is represented as

$$v_i = \sum_{j=(i-1)\times c+1}^{(i-1)\times c+k+1} s_{ij} sum(\mathbf{h}_{ij}), \tag{13}$$

Algorithm 2 Online Vector Boosting (OVB)

Input: Feature templates $\{\mathbf{z}_{ij}^{+}, \mathbf{z}_{ij}^{-}, j = 1, \ldots, n\}$.

1. Update n weak classifiers $\mathbf{h}_{ij}, j = 1, \ldots, n$ according to the strategy introduced by Section 3.3
2. Initialize $\mathbf{H}_{ij} = \mathbf{0}$ for all i, j
3. **for** $j = 1$ to k **do**
4. **for** $m = 1$ to n **do**
5. $\mathcal{J}^m = \mathcal{J}(\mathbf{H}_{ij} + \mathbf{h}_{im})$
6. **end for**
7. $m_j^{\star} = \arg\max_m \mathcal{J}^m$
8. $\mathbf{h}_{ij} \leftarrow \mathbf{h}_{im_j^{\star}}$
9. $\mathbf{H}_{ij} = \mathbf{H}_{ij} + \mathbf{h}_{ij}$
10. **end for**

Output: k selected feature templates $\{\mathbf{z}_{im_j^{\star}}^{+}, \mathbf{z}_{im_j^{\star}}^{-}, j = 1, \ldots, k\}$

where $sum(\mathbf{h}_{ij})$ can be efficiently computed by the integral images.

3.3. Online feature template update

CT [1] suffers drift when the target appearance changes much due to its fixed feature templates. In our algorithm, we proposes a conservative update scheme that only updates the templates with significant variations. Let $\Delta_{ij} = |\mathbf{h}_{ij}(\mathbf{p}_t) - \mathbf{h}_{ij}(\mathbf{p}_{t-1})|_2^1$ denote the corresponding template variations between two consecutive frames. If $\Delta_{ij} < \vartheta$, we keep the template $\mathbf{h}_{ij}$, otherwise, we update the template

$$\mathbf{h}_{ij}(\mathbf{p}_t) = \eta \mathbf{h}_{ij}(\mathbf{p}_t) + (1 - \eta)\mathbf{h}_{ij}(\mathbf{p}_{t-1}). \tag{14}$$

where $\eta > 0$ represents the update ratio.

3.4. Online trajectory rectification

Similar to CT [1], the tracking task is treated as a binary classification problem that the Naive Bayes classifier is adopted

$$H(\mathbf{v}) = \log\left(\frac{\prod_{i=1}^{c} p(v_i|y=+)p(y=+)}{\prod_{i=1}^{c} p(v_i|y=-)p(y=-)}\right) = \sum_{i=1}^{c} \log\left(\frac{p(v_i|y=+)}{p(v_i|y=-)}\right), \tag{15}$$

and the conditional distributions are assumed to be Gaussian distributed as

$$p(v_i|y=+) \sim \mathcal{N}(\mu_i^+, \sigma_i^+), p(v_i|y=-) \sim \mathcal{N}(\mu_i^-, \sigma_i^-), \tag{16}$$

where μ_i^+ and σ_i^+ are the mean and standard deviation of the i-th positive feature, respectively and similar to μ_i^- and σ_i^-. The parameters μ_i^+ and σ_i^+ are incrementally update by

$$\begin{aligned} \mu_i^+ &\leftarrow \lambda\mu_i^+ + (1-\lambda)\mu^+, \\ \sigma_i^+ &\leftarrow \sqrt{\lambda(\sigma_i^+)^2 + (1-\lambda)(\sigma^+)^2 + \lambda(1-\lambda)(\mu_i^+ - \mu^+)^2}, \end{aligned} \tag{17}$$

where $0 < \lambda < 1$ is a learning parameter, $\sigma^+ = \sqrt{\frac{1}{n^+}\sum_{k=0|y=+}^{n^+-1}(v_i(k)-\mu^+)^2}$ and $\mu^+ = \frac{1}{n^+}\sum_{k=0|y=+}^{n^+-1} v_i(k)$, n^+ is the number of positive samples. Parameters μ_i^- and σ_i^- are updated with similar rules.

When $conf = \max_{\mathbf{v}} H(\mathbf{v}) < \Theta$, which means the maximum classifier response is determined by the negative samples, the templates stop update. Then, we utilize the motion status in the former consecutive frames to predict the object location

$$\mathbf{l}_t = \mathbf{l}_{t-\Delta t} + \overrightarrow{\mathbf{v}_t}\Delta t, \tag{18}$$

where $\overrightarrow{\mathbf{v}_t} = \frac{\mathbf{l}_{t-1}-\mathbf{l}_{t-4}}{3}$ is the average motion velocity estimated from the former three frames, and $\Delta t = 1$ is the time step.

4. Experiments

4.1. Setup

Dataset: We evaluate the proposed algorithm on the CVPR2013 tracking benchmark [21] that includes results of 29 tracking algorithms on 50 fully annotated sequences ($\sim$ 26000 frames). For better evaluation and analysis of the strength and weakness of the tracking algorithms, the sequences are categorized according to 11 attributes, including illumination variation (IV), scale variation (SV), occlusion (OCC), deformation (DEF), motion blur (MB), fast motion (FM), in-plane rotation (IPR), out-of-plane rotation (OPR), out-of-view (OV), background clutters (BC), and low resolution (LR).

Parameter setting: The number of bags is set to $c = 150$. The number of templates in each bag is set to $n = 30$, in which $k = 5$ templates are selected. Threshold of the classifier score $\Theta = 0$. Threshold of appearance update is set to $\vartheta = 100$. The radius of searching window $\gamma = 25$. The

radius of sampling positive samples $\alpha = 2$, where $n^+ = 40$ positive samples are extracted. The inner radius of sampling negative samples $\zeta = 4$ while its corresponding outer radius $\beta = 15$, where $n^- = 40$ negative samples are selected. The update ratio of feature template $\eta = 0.05$, and the learning parameter of the classifier is set to $\lambda = 0.85$.

Evaluation metric: We employ the precision plot and success plot defined in [21] to evaluate the robustness of the tracking algorithms. The precision plot shows the percentage of frames whose estimated average center location errors are within the given threshold distance to the ground truth, in which the average center location error is defined as the average Euclidean distance between the center location of the tracked target and the manually labeled ground truth. The score at the threshold 20 pixels is defined as the precision score. Success plot shows the percentage of successful frames at the threshold ranging from 0 to 1. The successful frame is defined as the overlap score more than a fixed value, where the overlap ratio is defined as $S = \frac{\|r_t \bigcap r_a\|}{\|r_t \bigcup r_a\|}$ with the tracking output bounding box r_t and the ground truth bounding box r_a. For fair evaluation, the area under curve (AUC) is preferred to measure the success ratio. The one-pass evaluation (OPE) based on the average precision and the success rate given the ground truth of the first frame is used to evaluate the robustness of our algorithm.

4.2. Quantitative comparisons

Overall performance: Figure 5 illustrates overall performance of the top 10 evaluated tracking algorithms (i.e., SCM [6], Struck [20], TLD [11], ASLA [29], CXT [30], VTS [9], VTD [8], CSK [22], LSK [31], and LOT [32]) and the CT algorithm [23] in terms of precision plot and success plot. The proposed ACT ranks 1st in terms of both precision plot and success plot: the precision score of ACT is 0.714, which outperforms Struct [20] (0.656); meanwhile, in the success plot, the proposed ACT achieves the score of AUC 0.504, which outperforms SCM [29] by 0.5%. Note that the proposed ACT exploits only simple Haar-like features to represent the object and background, in which the simple naive Bayesian classifier is adopted with low computational complexity, yet it outperforms Struct and SCM that resort to complicate learning techniques in terms of both accuracy and efficiency.

Attribute-based performance: To facilitate analyzing strength and weakness of the proposed algorithm, we further evaluate ACT on videos with 11 attributes. Since the AUC score of the success plot is more accurate

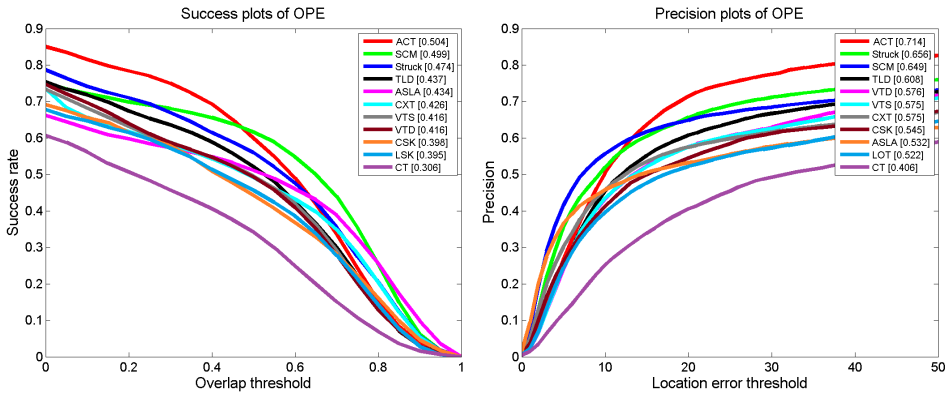

Figure 5: The success plots and precision plots of OPE for the top 10 trackers and CT. The performance score for each tracker is shown in the legend. The performance score of precession plot is at the error threshold of 20 pixels while the performance score of success plot is the AUC value. Best viewed on color display.

than that at the representative threshold (e.g., 20 pixels) of the precision plot, in the following we mainly analyze ACT based on the success plot.

Figure 6 shows that the success plots of videos with attributes that our method achieves favorable results, in which ACT ranks within top 2 on 8 out of 11 attributes. For the videos with attributes such as FM, MB, IV, DEF, BC, IPR, and OPR, ACT ranks 1st among all evaluated algorithms. In sequences with FM and MB, Struct ranks 2rd, showing that the tracker with wide range search window and dense sampling can perform well on these attributes, and so does ACT that sets search window size based on target size which prevents wrongly updating classifier from distracters. In sequences with IV and OCC, both SCM and ACT perform favorably well because they employ local features, in which ACT exploits the Haar-like features from the target via templates with varying sizes while SCM learns the local patch features from the target and background with sparse representation. Furthermore, both of them utilize the target template from the first frame, which is robust to drift problem. Similarly, on the OPR and IPR subsets, besides our tracker, the SCM and ASLSA trackers are also able to obtain satisfactory results, which can be attributed to the effective spare representations of local patches.

Figure 7 shows that ACT cannot perform well with three attributes, such as LR, OV, and SV. The low resolution makes ACT less effective to extract useful information from the target object. This can be improved by consider-

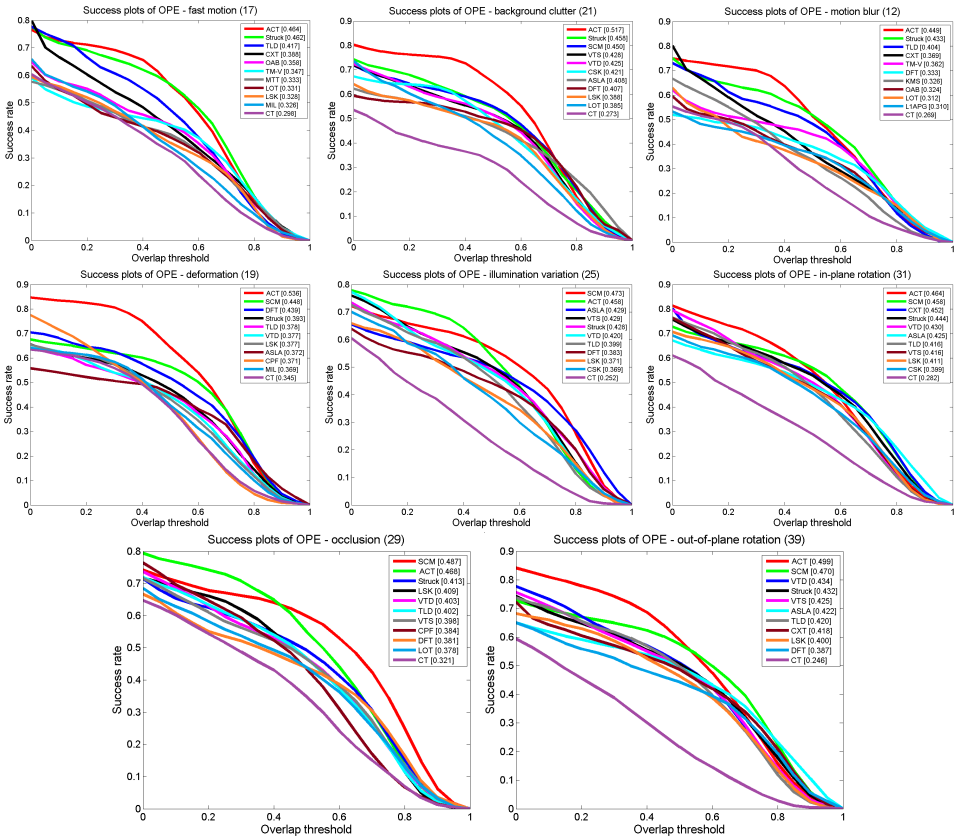

Figure 6: The success plots of videos with different attributes that ACT can achieve favorable results (within the top 2). Best viewed on color display.

ing the context information surrounding the target as [33]. Furthermore, the target appearance change drastically when OV occurs, and thus ACT cannot deal with these drastic variation favorably. However, the tracking failure in this case can be well reduced by memorizing information from some former frames and enlarging the search range. Finally, ACT only considers single scale tracking for simplicity, which can be readily extended to the multi-scale tracking by constructing scale pyramids, thereby improving performance on the sequences with SV attribute.

4.3. Qualitative comparisons

Deformation: Figure 8 shows the tracking results of three challenging sequences with deformation attributes. In the *Basketball* sequence, the target

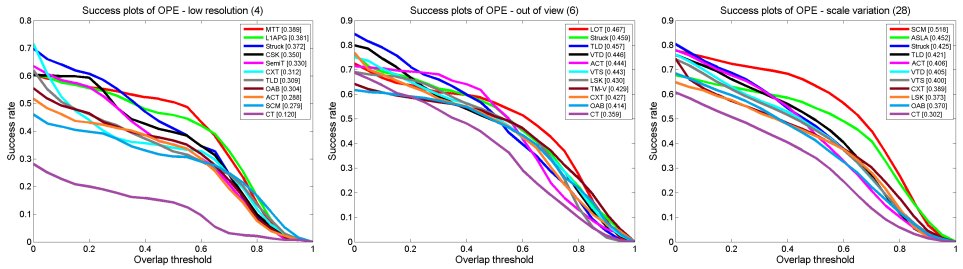

Figure 7: The success plots of videos with different attributes that ACT cannot perform well (outside the top 2). Best viewed on color display.

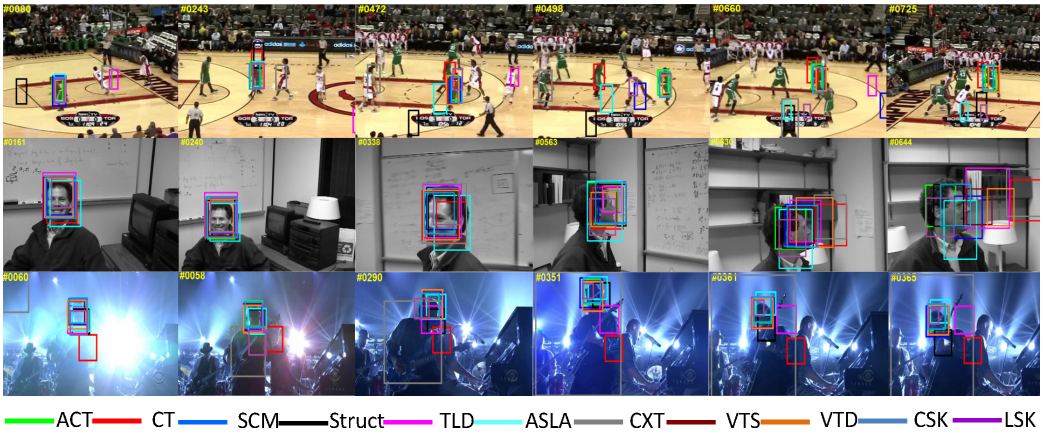

Figure 8: Qualitative results of the 11 trackers over sequences *Basketball*, *Fleetface* and *Shaking* from top to bottom (best viewed on high-resolution display). Object appearances therein change drastically with deformation.

undergos great changes as the player runs around, especially interferences from other plays. We observe that Struct, CXT, and TLD drift once other players hide the target (e.g., #34). The SCM, ASLA, CT, CSK and LSK drift when the object appearance begins to vary (e.g., #472). VTD, VTS and our ACT method are able to track the target in the whole sequence successfully. Our tracker can deal with deformation well due to its online appearance update and trajectory rectification strategies.

The *Fleetface* sequence suffers from scale changes, appearance varies, and background distraction when the object walks around the room. Many methods fail to track the object when the object turns his head, which results in dramatically appearance changes. Challenges also come from the interference caused by bookshelf, because the color and texture information is similar to

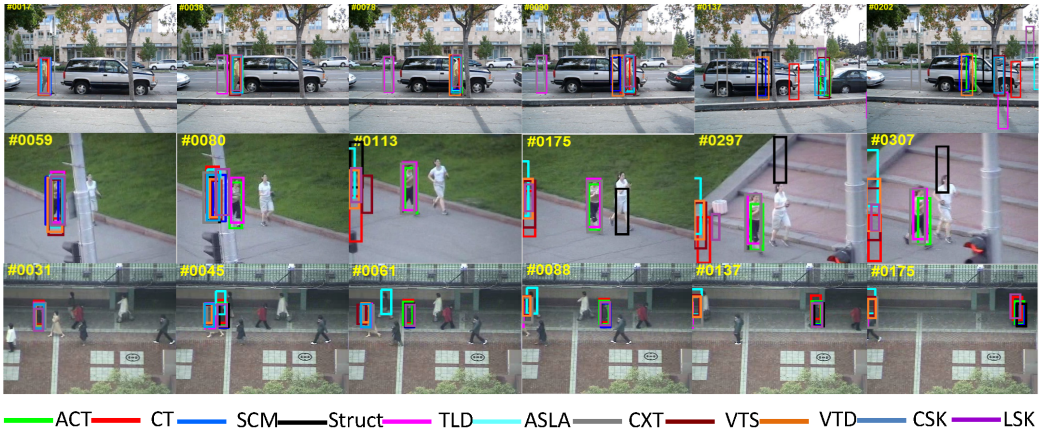

Figure 9: Qualitative results of the 11 trackers over sequences *David3*, *Jogging* and *Subway* from top to bottom (best viewed on high-resolution display). Object appearances therein changes drastically with heavy occlusion.

object at that time. ASLA, Struct and our ACT methods achieve well performance on this sequence.

In the *Shaking* sequence, the object undergoes both illumination change and pose variation. CSK, SCM, and VTD are able to track the object in this sequence, but with a lower success rate than our method.

Heavy occlusion: The targets in the sequences of Figure 9 undergo heavy occlusions from other objects. Furthermore, the targets in these sequences suffer from severe pose variations when the pedestrians turn round. Both make these sequences much challenging. Overall, ACT shows favorable performance to tackle these challenges, which attributes to the adaptive appearance model and online template update mechanism. When the confidence score of ACT decreases greatly to zero, the classifier and template stop updating, which can well prevent the tracker from drifting due to adding inaccurate samples.

Illumination change: Figure 10 shows the tracking results of three challenging sequences where the targets suffer from drastic illumination changes. In the *CarDark* sequence, a car runs along the street at night that suffers from large changes in environmental illumination and background clutters, and CT, TLD, CXT, and VTD drift gradually (e.g., #287). In contrast, SCM, Struct, ASLA, VTS, CSK, LSK and our ACT achieve much better performance. For the *Singer2* sequence, there is small contrast between the object and background besides illumination changes. Many trackers drift to

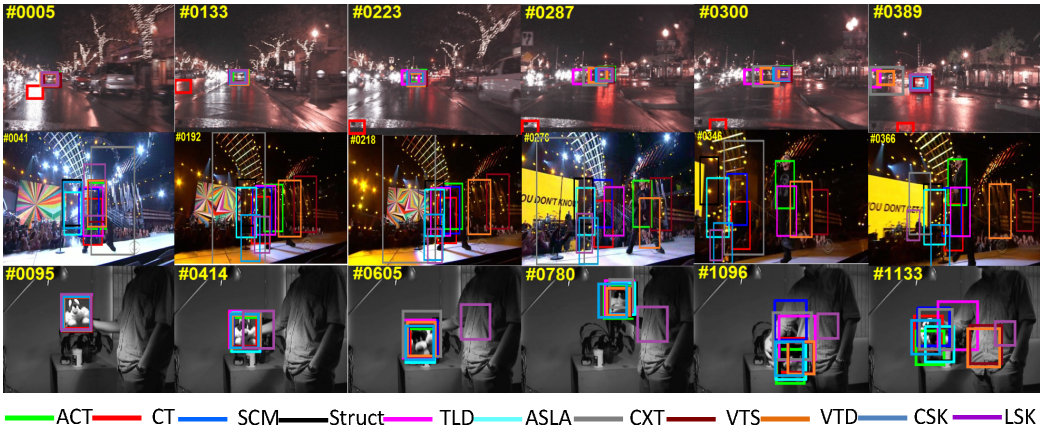

Figure 10: Qualitative results of the 11 trackers over sequences *CarDark*, *Singer2* and *Sylvester* from top to bottom (best viewed on high-resolution display). Objects therein undergo illumination changes.

the background at the beginning of this sequence when stage light changes drastically (e.g., #41). For the *Sylvester* sequence, challenges like IV, OPR, and IPR make it difficult to robustly track. Notwithstanding, our tracker achieve favorable performance due to its adaptive local appearance model.

Other challenges: Figure 11 presents the tracking results in which many other challenges occur in these sequences, such as MB, FM, BC, SV, etc. In the *Boy* sequence, a boy jumps regularly, causing MB and SV in his face, making it difficult to track. Our ACT performs well in this sequence because of online feature template update. The target in the *Deer* sequence suffers from MB, FM and BC, our ACT works well due to its online trajectory rectification strategy that can prevent the model update from inaccurate samples.

4.4. Analysis of ACT

Online feature template update (OFTU): To verify the effectiveness of OFTU, we develop a tracker named ACT-OFTU that removes the component of OFTU in ACT. The quantitative results are illustrated in Figure 12, where we can observe that ACT achieves much better performance than ACT-OFTU. OFTU emphasizes the importance of object appearance variance over time, where the stable templates are preserved. Furthermore, the update part in the templates takes into account the appearance variations, which can well adapt the target appearance variations over time.

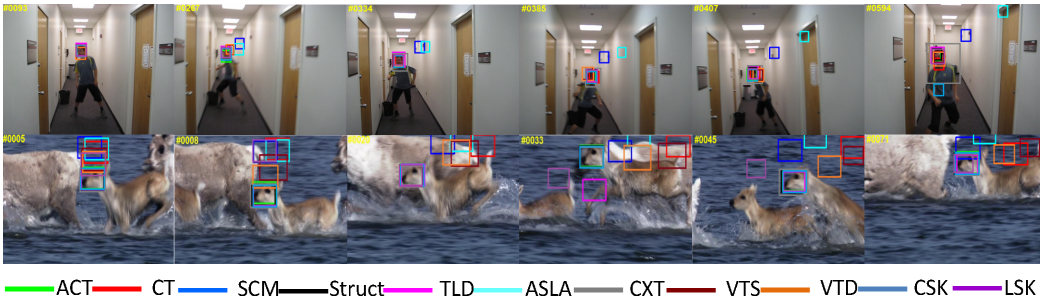

Figure 11: Qualitative tracking results of the 11 trackers over sequences *boy*, *Deer* from top to bottom (best viewed on high-resolution display. Objects therein undergo other challenges.

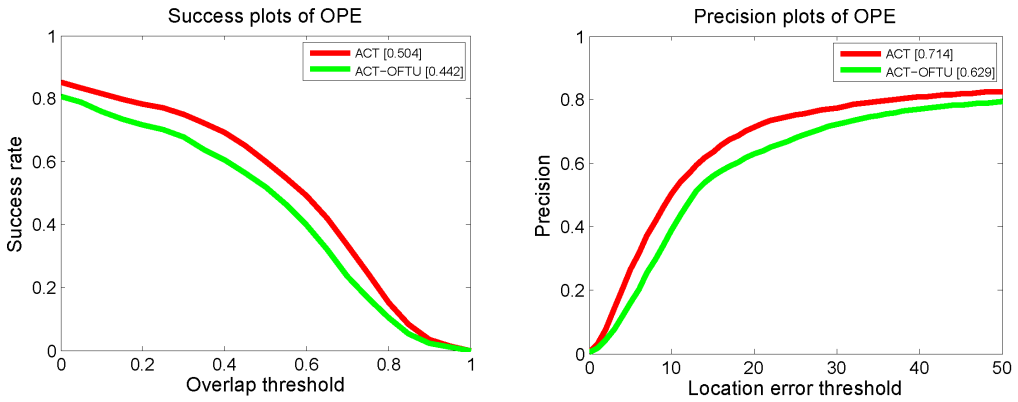

Figure 12: The success plots and the precision plots for ACT and ACT-OFTU

Online trajectory rectification (OTR): We design a tracker called ACT-OTR to justify the effectiveness of OTR in ACT. Figure 13 illustrates the quantitative results, where ACT outperforms ACT-OTR by a large margin, demonstrating the effectiveness of OTR that can well prevent the model update from inaccurate samples.

Online vector boosting feature selection (OVBFS): To justify the effectiveness of OVBFS, we construct a tracker named CT+OFTU+OTR that replaces the OVBFS component in ACT with the compressive Haar-like features in CT [23]. The quantitative results are shown in Figure 14, where CT performs unfavorably due to the fact that the feature templates may select noninformative features when they fall into the textureless regions, but with the OFTU and OTR, CT improves its performance significantly,

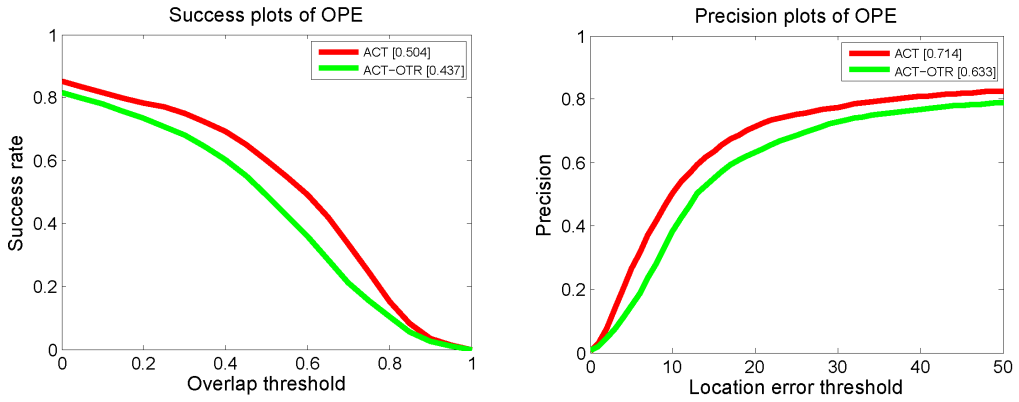

Figure 13: The success plots and the precision plots for ACT and ACT-OTR

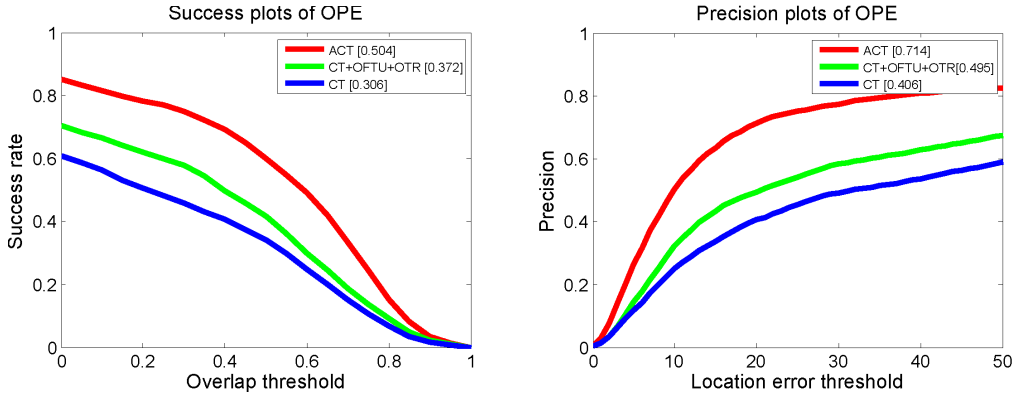

Figure 14: The success plots and the precision plots for CT, CT+OFTU+OTR, and ACT

which demonstrates the effectiveness of OFTU and OTR in ACT.

5. Conclusions

In this paper, we have proposed a novel adaptive compressive tracking algorithm that improves the CT algorithm [1] by a significantly large margin on the CVPR2013 tracking benchmark [21]. The proposed algorithm mainly includes three components: First, a novel vector boosting feature selection strategy has been proposed to design an effective appearance model; Second, a simple conservative model update strategy has been adopted that can preserve the stable information while filtering out the noisy appearance variations during tracking; Third, a simple and effective trajectory rectification

strategy has been developed that can refine the tracking location when possible inaccurate tracking occurs. Extensive evaluations on the CVPR2013 tracking benchmark have demonstrated the superior performance of the proposed algorithm over other state-of-the-art tracking algorithms.

References

[1] Kaihua Zhang, Lei Zhang, and Ming-Hsuan Yang. Real-time compressive tracking. In *Proceedings of European Conference on Computer Vision*, pages 864–877. 2012.

[2] Xi Li, Weiming Hu, Chunhua Shen, Zhongfei Zhang, Anthony Dick, and Anton Van Den Hengel. A survey of appearance models in visual object tracking. *ACM Transactions on Intelligent Systems and Technology*, 4(4):58, 2013.

[3] Alper Yilmaz, Omar Javed, and Mubarak Shah. Object tracking: A survey. *ACM Computing Surveys*, 38(4):13, 2006.

[4] Helmut Grabner, Michael Grabner, and Horst Bischof. Real-time tracking via on-line boosting. In *Proceedings of British Machine Vision Conference*, volume 1, page 6, 2006.

[5] Helmut Grabner, Christian Leistner, and Horst Bischof. Semi-supervised on-line boosting for robust tracking. In *Proceedings of European Conference on Computer Vision*, pages 234–247. 2008.

[6] Xu Jia, Huchuan Lu, and Ming-Hsuan Yang. Visual tracking via adaptive structural local sparse appearance model. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 1822–1829, 2012.

[7] Tianzhu Zhang, Bernard Ghanem, Si Liu, and Narendra Ahuja. Robust visual tracking via multi-task sparse learning. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 2042–2049, 2012.

[8] Junseok Kwon and Kyoung Mu Lee. Visual tracking decomposition. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 1269–1276, 2010.

[9] Junseok Kwon and Kyoung Mu Lee. Tracking by sampling trackers. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1195–1202, 2011.

[10] Boris Babenko, Ming-Hsuan Yang, and Serge Belongie. Robust object tracking with online multiple instance learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(8):1619–1632, 2011.

[11] Zdenek Kalal, Jiri Matas, and Krystian Mikolajczyk. Pn learning: Bootstrapping binary classifiers by structural constraints. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 49–56, 2010.

[12] Amit Adam, Ehud Rivlin, and Ilan Shimshoni. Robust fragments-based tracking using the integral histogram. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 798–805, 2006.

[13] David A Ross, Jongwoo Lim, Ruei-Sung Lin, and Ming-Hsuan Yang. Incremental learning for robust visual tracking. *International Journal of Computer Vision*, 77(1-3):125–141, 2008.

[14] Xue Mei and Haibin Ling. Robust visual tracking using ℓ_1 minimization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1436–1443, 2009.

[15] Chenglong Bao, Yi Wu, Haibin Ling, and Hui Ji. Real time robust l1 tracker using accelerated proximal gradient approach. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 1830–1837, 2012.

[16] Dong Wang, Huchuan Lu, and Ming-Hsuan Yang. Least soft-threshold squares tracking. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 2371–2378, 2013.

[17] Shai Avidan. Support vector tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(8):1064–1072, 2004.

[18] Robert T Collins, Yanxi Liu, and Marius Leordeanu. Online selection of discriminative tracking features. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(10):1631–1643, 2005.

[19] Kaihua Zhang and Huihui Song. Real-time visual tracking via online weighted multiple instance learning. *Pattern Recognition*, 46(1):397–411, 2013.

[20] Sam Hare, Amir Saffari, and Philip HS Torr. Struck: Structured output tracking with kernels. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 263–270, 2011.

[21] Yi Wu, Jongwoo Lim, and Ming-Hsuan Yang. Online object tracking: A benchmark. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 2411–2418, 2013.

[22] Joao F Henriques, Rui Caseiro, Pedro Martins, and Jorge Batista. Exploiting the circulant structure of tracking-by-detection with kernels. In *Proceedings of European Conference on Computer Vision*, pages 702–715. 2012.

[23] Kaihua Zhang, Lei Zhang, and M-H Yang. Fast compressive tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(10):2002–2015, 2014.

[24] Huihui Song. Robust visual tracking via online informative feature selection. *Electronics Letters*, 50(25):1931–1933, 2014.

[25] Chang Huang, Haizhou Ai, Yuan Li, and Shihong Lao. Vector boosting for rotation invariant multi-view face detection. In *Proceedings of the IEEE International Conference on Computer Vision*, volume 1, pages 446–453, 2005.

[26] Emmanuel J Candes and Terence Tao. Decoding by linear programming. *IEEE Transactions on Information Theory*, 51(12):4203–4215, 2005.

[27] Emmanuel J Candes and Terence Tao. Near-optimal signal recovery from random projections: Universal encoding strategies? *IEEE Transactions on Information Theory*, 52(12):5406–5425, 2006.

[28] Jerome Friedman, Trevor Hastie, Robert Tibshirani, et al. Additive logistic regression: a statistical view of boosting. *The annals of statistics*, 28(2):337–407, 2000.

[29] Wei Zhong, Huchuan Lu, and Ming-Hsuan Yang. Robust object tracking via sparsity-based collaborative model. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 1838–1845, 2012.

[30] Thang Ba Dinh, Nam Vo, and Gérard Medioni. Context tracker: Exploring supporters and distracters in unconstrained environments. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 1177–1184, 2011.

[31] Baiyang Liu, Junzhou Huang, Lin Yang, and Casimir Kulikowsk. Robust tracking using local sparse appearance model and k-selection. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 1313–1320, 2011.

[32] Shaul Oron, Aharon Bar-Hillel, Dan Levi, and Shai Avidan. Locally orderless tracking. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 1940–1947, 2012.

[33] Kaihua Zhang, Lei Zhang, Qingshan Liu, David Zhang, and Ming-Hsuan Yang. Fast visual tracking via dense spatio-temporal context learning. In *Proceedings of European Conference on Computer Vision*, pages 127–141. 2014.